Solving Multi-objective Dynamic Travelling Salesman Problems by Relaxation

Lorenzo A. Ricciardi
University of Strathclyde
Glasgow, United Kingdom
lorenzo.ricciardi@strath.ac.uk

Massimiliano Vasile
University of Strathclyde
Glasgow, United Kingdom
massimiliano.vasile@strath.ac.uk

ABSTRACT

This paper describes a method to solve Multi-objective Dynamic Travelling Salesman Problems. The problems are formulated as multi-objective hybrid optimal control problems, where the choice of the target destination for each phase is an integer variable. The resulting problem has thus a combinatorial nature in addition to being a multi-objective optimal control problem. The overall solution approach is based on a combination of the Multi Agent Collaborative Search, a population based memetic multi-objective optimisation algorithm, and the Direct Finite Elements Transcription, a direct method for optimal control problems. A relaxation approach is employed to transform the mixed integer problem into a purely continuous problem, and a set of smooth constraints is added in order to ensure that the relaxed variables of the final solution assume an integer value. A special set of smooth constraints is introduced in order to treat the mutually exclusive choices of the targets for each phase. The method is tested on two problems: the first is a motorised Travelling salesman problem described in the literature, the second is a space application where a satellite has to de-orbit multiple debris. For the first problem, the approach is generating better solutions than those reported in the literature.

CCS CONCEPTS

• **Mathematics of computing → Mixed discrete-continuous optimization**; *Nonconvex optimization*; • **Applied computing → Multi-criterion optimization and decision-making**; *Aerospace*;

KEYWORDS

Multi-objective Optimisation, Global Optimisation, Optimal control, Mixed Integer Nonlinear programming, Memetic Algorithms

ACM Reference Format:
Lorenzo A. Ricciardi and Massimiliano Vasile. 2019. Solving Multi-objective Dynamic Travelling Salesman Problems by Relaxation. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3319619.3326837

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3326837

1 INTRODUCTION

This paper proposes a method to solve multi-objective dynamic Travelling Salesman Problems. The goal is to find the trajectory of a vehicle that visits multiple targets minimising multiple conflicting objectives, where the order in which the targets are visited is not specified a priori but is free to be optimised.

The Travelling Salesman Problem, in its various forms, is one of the most studied in the field of optimisation. For this reason, even an extremely condensed overivew of the relevant literature is outside of the scope of this short paper. A distinctive aspect of the problem here investigated that sets it aside from the vast majority of the literature concerning the TSP is that the trajectory connecting a point to another is not an abstract edge connecting two nodes of a graph. Instead, an actual trajectory has to be computed accounting for a dynamical model of the vehicle of the Travelling Salesman. The dynamical model is expressed in terms of a general set of nonlinear ODEs and some free control parameters that can influence the trajectory of the system. The cost associated to an itinerary is thus not just a function of the sequence of the cities visited and eventually the times of the visits, but depends also on the time histories of the controls. This is more typical of the field of optimal control, where an optimal policy is sought to steer a dynamical system from an initial state to another while minimising some cost function. In this sense, the adjective *dynamic* TSP is due to the presence of a dynamical system, as opposed to an explicit time-varying nature of the objective function or of the connections between the edges of a graph. While the latter have been extensively investigated, for instance in [1, 2, 6, 7, 12, 13], the former has received less attention. This work can be seen as a multi-objective extension of the problem class tackled for instance in [3–5, 17]. References [4, 7, 17] show how the formulation of the dynamic (or motorised) TSP can be employed to generate space missions visiting multiple asteroids but all deal with single objective formulations.

The problem is here formulated as a multi-phase multi-objective hybrid optimal control problem where the final conditions of each phase depend on the choice of which target is visited. The solution method adopted in this work closely follows [8], which dealt with general multi-phase multi-objective optimal control problems. The approach is based on a direct transcription of the optimal control problem with Direct Finite Elements in Time (DFET) [9, 14] and a solution of the resulting multi-objective nonlinear programming (MONLP) problem with Multi Agent Collaborative Search (MACS) [10, 15, 18], a population based global memetic multi-objective optimisation algorithm.

In order to treat mixed integer problems, the method here proposed employs a relaxation approach for the integer variables, with

an associated set of smooth constraints to ensure that the final values of the relaxed variables is integer. An additional set of smooth constraints is introduced to tackle the combinatorial aspect of the problem, ensuring that the target destinations are visited exactly once.

Two numerical examples will show the validity of the overall approach: the first one is a motorised Travelling salesman problem with simple dynamics, initially proposed in [16]. This was chosen as it is easy to replicate and thus constitutes a good comparison problem. The second test case is an application consisting in the preliminary design of a space mission to de-orbit or repair malfunctioning satellites in Geostationary orbit.

2 PROBLEM FORMULATION

Multi-objective Hybrid Optimal Control problems can be formulated as:

$$\begin{aligned} &\min_{\mathbf{u}\in U,\boldsymbol{\sigma}\in\sigma,\mathbf{b}\in B,\boldsymbol{\mu}\in M} \mathbf{J} \\ &s.t. \\ &\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t) \\ &\mathbf{g}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t) \geq 0 \\ &\boldsymbol{\psi}(\mathbf{x}(t_0),\mathbf{x}(t_f),\mathbf{u}(t_0),\mathbf{u}(t_f),\mathbf{b},\boldsymbol{\mu},t_0,t_f) \geq 0 \\ &t \in [t_0,t_f] \end{aligned} \tag{1}$$

where $\mathbf{J} = [J_1, J_2, ..., J_i ..., J_m]^T$ is, in general, a vector of objectives J_i that are functions of the state vector $\mathbf{x} : [t_0, t_f] \rightarrow \mathbb{R}^{n_x}$, continuous control variables $\mathbf{u} \in L^{\infty}(U \subseteq \mathbb{R}^{n_u})$, discrete control variables $\boldsymbol{\sigma} \in \sigma \subseteq \mathbb{Z}^{n_\sigma}$, continuous static parameters $\mathbf{b} \in B \subseteq \mathbb{R}^{n_b}$, discrete static parameters $\boldsymbol{\mu} \in M \subseteq \mathbb{Z}^{n_z}$ and time t. The functions $\mathbf{x}(t)$ belong to the Sobolev space $\mathcal{W}^{1,\infty}$ while the objective functions are $J_i : \mathbb{R}^{3n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\sigma} \times \mathbb{Z}^{n_z} \times \mathbb{R}^2 \longrightarrow \mathbb{R}$. The objective vector is subject to a set of dynamic constraints with $\mathbf{F} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\sigma} \times \mathbb{Z}^{n_z} \times [t_0, t_f] \longrightarrow \mathbb{R}^{n_x}$, algebraic constraints $\mathbf{g} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\sigma} \times \mathbb{Z}^{n_z} \times [t_0, t_f] \longrightarrow \mathbb{R}^{n_g}$, and boundary conditions $\boldsymbol{\psi} : \mathbb{R}^{2n_x} \times \mathbb{R}^{2n_u} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\sigma} \times \mathbb{Z}^{n_z} \times \mathbb{R}^2 \longrightarrow \mathbb{R}^{n_\psi}$, where $\mathbf{F}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t)$, $\mathbf{g}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t)$ and $\boldsymbol{\psi}(\mathbf{x}(t_0),\mathbf{x}(t_f),\mathbf{u}(t_0),\mathbf{u}(t_f),\mathbf{b},\boldsymbol{\mu},t_0,t_f)$ are vector fields.

When N_p phases are present, dynamic constraints, path and boundary constraints, and objective functions are defined on each timeline. In order to connect different timelines, a set of N_{ip} interphase constraints is introduced:

$$\psi_{s_p}\left(\mathbf{x}_{0,\mathbf{I}_{s_p}}, \mathbf{x}_{f,\mathbf{I}_{s_p}}, \mathbf{u}_{0,\mathbf{I}_{s_p}}, \mathbf{u}_{f,\mathbf{I}_{s_p}}, \mathbf{b}_{\mathbf{I}_{s_p}}, \boldsymbol{\mu}_{\mathbf{I}_{s_p}}, t_{0,\mathbf{I}_{s_p}}, t_{f,\mathbf{I}_{s_p}}\right) \geq 0 \tag{2}$$

with $s_p = 1, ..., N_{ip}$. The index vector $\mathbf{I}_{s_p}$ collects all the indexes of the phases that are connected by constraint ψ_{s_p}. Note that the number of phases is fixed, but their temporal order is actually defined by the inter-phase constraints (2).

The problem can be discretised following the procedure for DFET transcription. This results in a Multi Objective Mixed Integer Nonlinear Programming (MOMINLP) problem coming from the transcription of problem (1), with the inclusion of interphase constraints (2). In vector form it can be written as:

$$\begin{aligned} &\min_{\mathbf{y}\in Y,\mathbf{p}\in\Pi,\mathbf{d}\in D} \tilde{\mathbf{J}} \\ &s.t. \\ &\mathbf{C}(\mathbf{y},\mathbf{p},\mathbf{d}) \geq 0 \end{aligned} \tag{3}$$

where $\mathbf{y}$ collects all the discretised state variables, $\mathbf{p}$ collects all the static and discretised dynamic control variables, $\mathbf{d}$ collects all the integer valued control variables and static parameters, and $\mathbf{C}$ collects all constraints, including boundary and interphase conditions.

3 SOLUTION OF THE MIXED INTEGER OPTIMAL CONTROL PROBLEM

To solve Multi-Objective Optimal Control problems, the following approach was proposed in [8]: first, the multi-objective optimal control problem (1) is translated into a multi-objective NLP problem by using the DFET transcription scheme, which discretises the dynamics of the problem and converts it the finite dimensional problem (3). An automatic and unsupervised process generates feasible guesses before the main loop of the optimisation starts. The multi-objective NLP is then solved by a modified version of the MACS optimisation algorithm, employing two different and complementary formulations. In the following section, a brief description of the MACS algorithm is given, followed by a brief description of their coupling strategy.

3.1 Description of the MACS algorithm

MACS is a memetic multi-objective optimisation algorithm in which a population of agents is initially seeded in the search space with a Latin Hypercube sampling. All agents can perform a set of individual actions to explore their neighborhood and improve their position. A user defined number of agents, called social agents, is associated with a weight vector. Weight vectors are generated to be uniformly spread in the unit sphere on ${\mathbb{R}_+}^m$, where $\mathbb{R}_+$ is the set of non negative real numbers. These weight vectors are used to compute the Chebychev scalarisation of the solution vector associated to that agent. A trial solution with a better Chebychev scalarisation criterion is considered to improve the current solution. The dominance criterion is also applied in order to detect an improvement.

The individualistic actions that each agent can perform are Pattern Search, Differential Evolution and Inertia. For the Pattern Search heuristic, one randomly chosen optimisation variable is perturbed by a random amount within the local neighbourhood defined around each agent. If no improvement is made, another trial is attempted in the opposite direction, and the process is then repeated with another variable until a dynamically adjusted maximum number of directions is scanned or an improvement is made. If Pattern Search does not find an improved solution, a Differential Evolution step takes place choosing three other different agents. If at a previous iteration a successful improvement direction was found, the Inertia action takes place before the Pattern Search: another attempt is made in the same successful direction, with a random step size. If no improvement is made by any action, the local neighbourhood associated to that agent is reduced by a user defined factor, while it is increased by the same factor if a successful step is made. The initial size of the local neighbourhood is the entire search space, and it is also the maximum size it is allowed to assume. It's minimum size is instead defined by the user, who chooses the maximum number of contractions of the local neighbourhood.

Non-dominated solutions are saved to an external archive, with a user defined maximum size. A special archiving strategy ensures a good distribution of points across the Pareto front.

After the individual actions are performed and the archive has been updated, Social agents can then perform social actions, exploiting the information coming from the archive or from other agents to collectively advance towards the Pareto front. Social actions employ again the Differential Evolution heuristic, but this time the choice of the other 3 solutions can come from either the archive or the population. All solutions are again evaluated for improvement and eventually added to the archive. Social agents are then moved to the position in the archive that best satisfies their Chebychev scalarisation criterion.

Individual and social actions are repeated until a maximum number of function evaluations is reached. For a more detailed explanation of the MACS algorithm, the interested reader is invited to read [10].

3.2 Description of the formulations

Two different and complementary formulations are employed to solve problem (3). The first one consists of a further reformulation of the multi-objective NLP into a bi-level optimisation problem. At the outer level, MACS uses its heuristics to generate trial solution vectors. This trial solution vector is received by the inner level, which uses it as a first guess to enforce the constraints through an NLP solver. Since the outer level always modifies fully feasible solutions and only changes the control variables leaving the state variables unaltered, the solution of each inner level NLP is very fast, typically requiring only a handful of iterations. Once the NLP solver converges, the solution is passed back to the outer level, which evaluates the objective functions and employs the dominance criterion and the Chebychev scalarisation to decide weather the new solution is better than the existing ones. This bi-level formulation allows to strictly solve nonlinear equality and inequality constraints, provides global exploration capabilities and a good spreading of the solutions on the Pareto front.

In order to guarantee local optimality of the solution, a single level gradient based refinement step takes place every given number of iterations. This refinement step operates on a smooth single level reformulation of the problem. The reformulation operates on a modified Pascoletti-Serafini scalarisation of the initial multi-objective optimisation problem and allows a simultaneous handling of feasibility and local optimality. Very importantly, the Chebychev scalarisation criterion used in the bi-level approach is consistent with the Pascoletti-Serafini scalarisation employed in the single level approach. The method can thus seamlessly transition between a global exploration mode that generates evenly spread solutions on the Pareto front, to a local exploration mode able to guarantee the local optimality of all the solutions along the same descent directions in criteria space that were used by the global exploration mode. A more complete description of this approach is described in [8].

This approach can be further extended in order to treat mixed-integer problems. For the outer level, the heuristics of MACS can operate both with discrete and continuous variables, thus no modification is required. For the inner level and for the single level gradient based approach instead the discrete variables were relaxed in order to work with the NLP solver. In order to ensure that the final values of the relaxed variables are integer, an additional set of constraints is imposed. This way the inner level is able to modify the value of the relaxed variables in order to get feasible solutions. Algorithm 1 summarises the overall solution strategy. The following subsection explains in greater detail the constraints imposed to ensure that the relaxed variables assume an integer value.

Algorithm 1 MACS optimal control (MACSoc) framework

1: Initialise population $\mathcal{P}_0$ and global archive $\mathcal{A}_0$, $k = 0$, $\rho_{\mathcal{B}} = 1$
2: Initialise weight vectors $\boldsymbol{\omega}$
3: **while** $n\_fun\_eval < max\_fun\_eval$ **do**
4: Run individualistic heuristics on the relaxed problem using bi-level formulation Algorithm 2
5: $\mathcal{P}_k \rightarrow \mathcal{P}_k^+$
6: Update archive $\mathcal{A}_k$ with potential field filter
7: Run social heuristics combining $\mathcal{P}_k^+$ and $\mathcal{A}_k$ using bilevel formulation Algorithm 2
8: Update archive $\mathcal{A}_k$ with potential field filter
9: $\mathcal{P}_k^+ \rightarrow \mathcal{P}_k^\dagger$
10: **if** local search triggered **then**
11: Run gradient based refinement using single level formulation
12: $\mathcal{P}_k^\dagger \rightarrow \mathcal{P}_k^*$
13: Update archive $\mathcal{A}_k$ with potential field filter
14: $\mathcal{P}_k^* \rightarrow \mathcal{P}_{k+1}$
15: **else**
16: $\mathcal{P}_k^\dagger \rightarrow \mathcal{P}_{k+1}$
17: **end if**
18: $k = k + 1$
19: Update $\rho_{\mathcal{B}}$
20: **end while**

Algorithm 2 Bi-level formulation

1: Apply chosen heuristic to modify controls and static variables
2: Solve feasibility problem with relaxed variables
3: Re-solve feasibility problem imposing (4)
4: Evaluate objectives of feasible solution

3.3 Relaxation and Integrality constraints

Within the inner level and the single level refinement the discrete variables were relaxed and treated as continuous. This will likely generate a solution where the relaxed variables do not assume integer values. After a relaxed solution is found, the NLP solver is invoked again but the following constraint is imposed in order to force it to converge to solutions where the relaxed variables assume integer values:

$$\sin(\pi\tilde{\sigma}_j) = 0 \tag{4}$$

where $\tilde{\sigma}_j$ is a relaxed variable.

This simple and smooth constraint is satisfied only for integer values of the relaxed variable x_j. However, since this constraint has many possible solutions, it can introduce several local optima.

For this reason it is enforced only after a solution to the relaxed problem is found. The delayed imposition of constraints (4) should allow the NLP solver to avoid the local minima introduced by this constraint and converge to the integer value closest to the relaxed solution.

3.4 Treatment of free-target boundary constraints

For the class of problems of interest in this paper, the boundary conditions of each phase can be expressed as:

$$\mathbf{x}(t_f; i) = \sum_j \mu_{i,j} \mathbf{P}_j \tag{5}$$

where $\mathbf{x}(t_f; i)$ indicates the boundary conditions at the end of phase i, $\mu_{i,j}$ are static optimisation variables which can only assume a value of 0 or 1, and $\mathbf{P}_j$ are the target boundary conditions. Variables $\mu_{i,j}$ can be considered as a choice on the target associated to each phase. In facts, if $\mu_{1,j} = 1$ while all other $\mu_{i,j} = 0$ the constraint will impose phase j to terminate at the position of target P_1. However, exactly one $\mu_{i,j}$ per phase must be equal to one. In addition, it is required to visit each target exactly once. This particular set of mutual exclusivity constraints needs a special treatment.

In [16] it is proposed to arrange the discrete variables $\mu_{i,j}$ in a matrix, relax the variables $\mu_{i,j}$ into $\tilde{\mu}_{i,j}$ and impose

$$\begin{cases} \sum_i \tilde{\mu}_{i,j} = 1 \ \forall j = 1 \cdots n \\ \sum_j \tilde{\mu}_{i,j} = 1 \ \forall i = 1 \cdots n \end{cases} \tag{6}$$

where $\tilde{\mu}_{i,j}$ is the relaxed i^{th} discrete variable for phase j. For an n targets problem, this results in a total of $2n$ constraints. The idea behind this set of constraints is to ensure that, if the variables assume a value of 0 or 1, only one element per row and column can be 1, thus resulting in a unique choice of target per each phase. These constraints were then employed to obtain a relaxed solution which constituted the upper bound for a branch and bound method: one relaxed variable was then split into a branch where it had a fixed value of 0 and a value of 1 in the other. The branch and bound method progressed until all relaxed variables were given a value of either 0 or 1, progressively discarding regions of the search space which were considered unpromising.

This method has the advantage of not requiring any further special treatment for the discrete variables, which are treated separately to the continuous ones. However, it has two main disadvantages: the first is that in order to obtain a solution to the full non-relaxed problem it has to solve many NLPs, in the best case equal to twice the number of binary variables and in the worst case equal to performing a full enumeration of the 2^n possible combinations. The second, less obvious disadvantage, is that the method relies on the NLP to provide an upper bound for the solution. If the problem has multiple local solutions, the bounds computed by the NLP might be incorrect and the method might discard the region of the search space where the true optimal solution lays.

The set of constraints (6) has two major problems when applied with the current framework: the first one is that it is composed by $2n$ equality constraints. When adding constraints (4) to enforce each $\tilde{\mu}_{i,j}$ to assume a value of 0 or 1, the problem becomes overdetermined for the $\tilde{\mu}_{i,j}$, given that constraints (4) already impose an equality constraint per relaxed variable. Even if the constraints are redundant at the solution points, the NLP solver might have serious difficulties in handling this situation. A second less obvious problem is that only one $\tilde{\mu}_{i,j}$ involved in each of constraints (6) should be greater than 0.5, before applying constraints (4). The reason for this is that, in order to satisfy the constraints, the NLP solver performs a linearisation of the constraints. Thus, it will try to satisfy the constraints by pushing each $\tilde{\mu}_{i,j}$ towards either 0 or 1 depending on weather each $\tilde{\mu}_{i,j}$ is lower or greater than 0.5. However, if more than one $\tilde{\mu}_{i,j}$ is greater than 0.5, or none of them are, it will be impossible for the NLP solver to satisfy simultaneously (4) and (6).

In order to prevent this problem, another approach is here proposed. First, equality constraints (6) are rewritten as inequality constraints

$$\begin{cases} \sum_i \tilde{\mu}_{i,j} \leq 1 + \epsilon & \forall j = 1 \cdots n \\ \sum_i \tilde{\mu}_{i,j} \geq 1 - \epsilon & \forall j = 1 \cdots n \\ \sum_j \tilde{\mu}_{i,j} \leq 1 + \epsilon & \forall i = 1 \cdots n \\ \sum_j \tilde{\mu}_{i,j} \leq 1 - \epsilon & \forall i = 1 \cdots n \end{cases} \tag{7}$$

where ϵ is a fixed positive parameter. This set of linear inequality constraints solves the issue of overdetermination of the system, because the only equality constraints remaining are from Eq. (4). Moreover it creates a feasible region of thickness 2ϵ around the hyperplane defined by the constraints (6), which should be easier to satisfy than a strict equality constraint. However, it is still possible that more than one of the $\tilde{\mu}_{i,j}$ is greater than 0.5, or none are. In order to solve this issue, an additional set of constraints is imposed:

$$\begin{cases} \sum_i \tilde{\mu}_{i,j}^{\alpha} \geq n \left(\frac{1}{2} + \epsilon\right)^{\alpha} & \forall j = 1 \cdots n \\ \sum_j \tilde{\mu}_{i,j}^{\alpha} \geq n \left(\frac{1}{2} + \epsilon\right)^{\alpha} & \forall i = 1 \cdots n \end{cases} \tag{8}$$

where n is the number of targets and $\alpha \geq \frac{\log_2(n)}{\log_2 \frac{2}{1+2\epsilon}}$. For this second set of constraints to work, ϵ should be lower than 0.5. For simplicity, the actual value of α used in the constraint can be rounded to the smallest even integer larger than the minimum value. Constraint (8) was derived from the following considerations: in principle it would be necessary to guarantee that the largest of the $\tilde{\mu}_{i,j}$ for each row and column is strictly greater than $\frac{1}{2} + \epsilon$. In that case, form constraints (7), the sum of all other $\tilde{\mu}_{i,j}$ would be strictly less than $\frac{1}{2}$, thus guaranteeing that all other $\tilde{\mu}_{i,j}$ are individually less than $\frac{1}{2}$. The maximum value of a vector is equivalent to the infinity norm of the vector. However, since the infinity norm is not a differentiable function, its usage would be problematic when included inside an NLP problem. To overcome that problem, it is sufficient to find a norm with an exponent high enough to obtain the same result, but still having a smooth behaviour and possibly without resulting in numerical issues. Equation (8) does exactly that: the minimum value of α was computed in such a way that the

vertices of the hypercube connected by an edge that stems from the origin are never outside of the bubble with the above given radius. In other words, if the minimum value of α is chosen, the constraints will be tangent to each permutation of the vector $(1, 0, 0, ..., 0)$. If a larger value of α is used, the smallest x_{min} for which any permutation of the vector $(x_{min}, 0, 0, ..., 0)$ satisfies Equation (8) is $x_{min} = \left(\frac{1}{2} + \epsilon\right) n^{\frac{1}{\alpha}}$. Since a positive root of a positive number greater than 1 is always greater than 1, the second term of the last equation is always greater than 1, thus the minimum possible $x_{min} > \frac{1}{2} + \epsilon$. The constraints thus behave as desired. Moreover the exponent α grows logarithmically with n, the number of targets, thus the approach should not encounter numerical issues even for large number of targets. The main limitation of the approach is indeed given by the maximum size of the problem that can be handled by the NLP solver.

It is important to remark that the feasible region of constraints (7) and (8) is disconnected and has a number of islands equal to the number of possible sequences. Thus, many different feasible and locally optimal solutions are possible and a global exploration approach is necessary.

4 TEST CASES

This section presents two test cases. Since the main goal of this paper is to perform first tests on the validity of the proposed approach, all the test cases are very simple in terms of combinatorial complexity. For the same reason, the first test case is is a multi-objective extension of a simple single objective problem already solved in the literature. This allows to directly compare the solutions obtained by the proposed approach with the known solutions.

In the first test case, referred to in the literature as a motirised Travelling Salesmen Problem, a vehicle has to visit several targets whose order is not specified. The second test case is a space application, where a spacecraft has to rendez-vous with multiple targets in order to deorbit or perform on-orbit servicing, and again the order in which the targets are visited is not given a priori.

The algorithm was implemented in Matlab® 2017b and run on a Linux workstation with 8 GB of RAM and an Intel i7-4790 cpu.

4.1 A motorised Travelling Salesmen Problem

This section describes a multi-objective extension of a problem presented in [16]: a vehicle, described by a simple two dimensional dynamic model, starts from the origin of the plane and has to visit three target destinations before finally returning to its starting position. It is controlled by the magnitude of the acceleration and by the steering rate, both of which are limited. The order in which the three targets are to be visited is not specified a priori but has to be found as part of the solution, which in the original reference had to minimise the total mission time. The vehicle has to pass on each target point without any restriction on the velocity or direction of velocity at the rendez-vous, while at the final time it had to be at rest at the original position. This problem can be seen as an extension of a classic Travelling Salesman Problem, where the presence of a dynamical model for the Salesmen significantly changes the nature of the problem and its complexity.

The objectives for this problem are the minimisation of the total time for the tour, and the minimisation of the energy required:

$$\min_{t_f, \mathbf{u}, \boldsymbol{\mu}} (J_1, J_2)^T = \left(t_f, \int_{t_0}^{t_f} u_1^2 dt\right) \tag{9}$$

The dynamics of the vehicle are given by:

$$\begin{cases} \dot{x} &= v\cos(\alpha) \\ \dot{y} &= v\sin(\alpha) \\ \dot{v} &= u_1 \\ \dot{\alpha} &= u_2 \end{cases} \tag{10}$$

where x and y denote the position of the vehicle on the plane, v the magnitude of its velocity and α the direction of the velocity vector. The vehicle is controlled by the magnitude of the acceleration u_1 and by the steering rate u_2, both of which are limited: $u_1^2 \leq 1$, $u_2^2 \leq 1$.

The vehicle starts at the origin at rest, and has to pass on 3 targets, whose location is

$$\begin{cases} P_1 = & (1, 2) \\ P_2 = & (2, 2) \\ P_3 = & (2, 1) \end{cases} \tag{11}$$

The problem was formulated as a 4 phase problem, with the last phase targeting the origin. Matching conditions were imposed between the states of the various phases, and final conditions for each phase were imposed as per Eq. (5). The problem was discretised using 3 DFET elements of order 7 for both states and controls and each phase, resulting in a problem with 638 variables. Bounds for the optimisation variables are: $-5\,\mathrm{m} \leq x \leq 5\,\mathrm{m}$, $-5\,\mathrm{m} \leq y \leq 5\,\mathrm{m}$, $-10\,\mathrm{m\,s^{-1}} \leq v \leq 10\,\mathrm{m\,s^{-1}}$, $-10\,\mathrm{rad} \leq \alpha \leq 10\,\mathrm{rad}$, $-1\,\mathrm{m/s^2} \leq u_1 \leq 1\,\mathrm{m/s^2}$, $-1\,\mathrm{rad\,s^{-1}} \leq u_2 \leq 1\,\mathrm{rad\,s^{-1}}$, $0\,\mathrm{s} \leq t_f \leq 15\,\mathrm{s}$.

The algorithm was run with 10 agents for a total of 40000 function evaluations with standard settings, and the single level gradient based refinement every 10 iterations. 10 solutions were kept into the archive. 30 independent runs were made, in order to gather some statistics about the algorithm. Figure 1 shows the Pareto front of all the 30 independent runs of the algorithm. As it can be seen, the problem has several local Pareto fronts. This was already stated in [16], who reported the presence of multiple locally optimal solutions for the single objective problem minimising tour time. The algorithm seems to converge most of the time on a local Pareto front that is not the global one, indicating that the local solutions are misleading and not easy to bypass, at least with the given number of function evaluations. Of all the 30 runs, only 2 managed to return entirely globally non dominated solutions. For future reference, the GD and IGD metrics of the Pareto fronts were computed, as reported in Table 1. Since no previous reference Pareto front is known for this problem, the reference Pareto front with which the metrics were computed was generated by taking the most uniformly spread 10 solutions among the globally non-dominated set using our archiving algorithm.

Figure 2 and 3 show the Pareto front and the trajectories computed for one run which returned globally non-dominated solutions. Lower time solutions require more energy, as expected.

The order in which the target locations are visited is the same for all the trajectories, starting from P_1 and proceeding to P_2 and

	Mean	Variance	% <= 0.1
IGD	0.1709	0.0060	10%
GD	0.1722	0.0091	10%

Table 1: Statistical values of the metrics of the 30 Pareto fronts

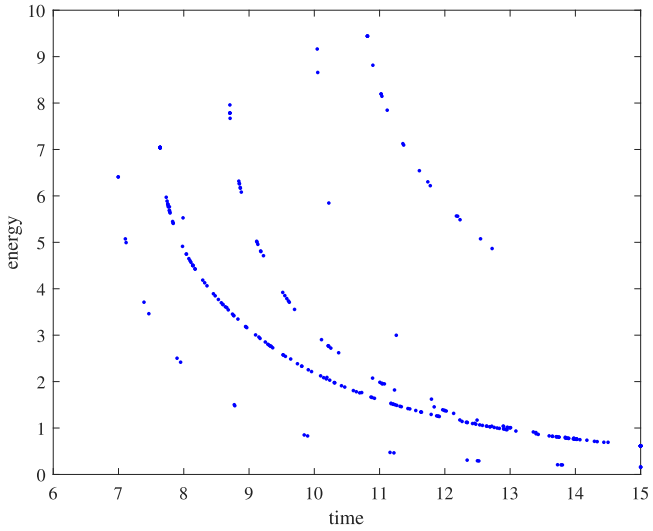

Figure 1: Pareto front for 30 independent runs on the motorised Travelling Salesman Problem

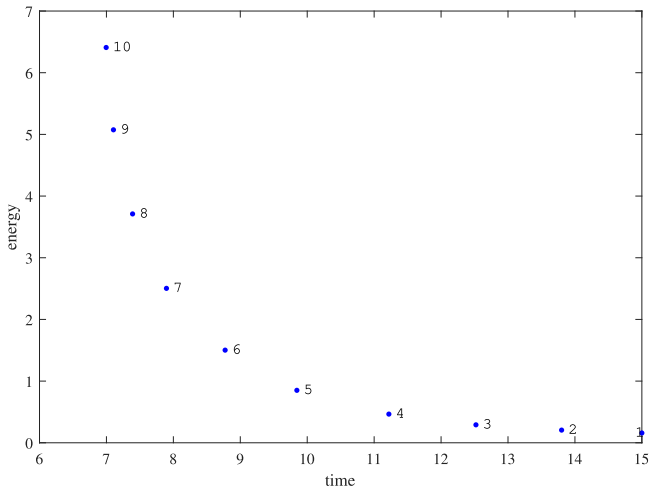

Figure 2: Pareto front for a completely non-dominated single run of the motorised Travelling Salesman Problem

P_3. The solutions found have a very reasonable shape, although lower time solutions follow longer curved trajectories while longer time solutions have shorter quasi rectilinear trajectories.

To explain this counter-intuitive result it is useful to look at Figure 5, which shows the time histories of the magnitude of the

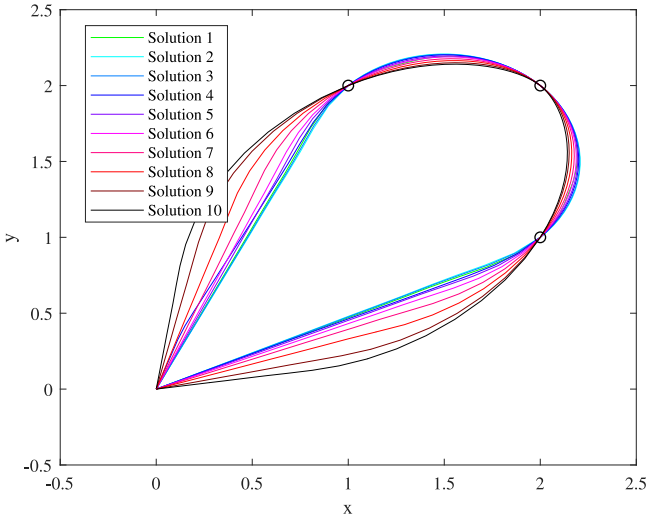

Figure 3: Trajectories for the motorised Travelling Salesman Problem

velocity and the corresponding control, the acceleration. As expected, the minimum time trajectories are the result of a bang-bang control profile for the accelerations. In this case, the bang-bang switch happens twice: a full acceleration is followed by a full deceleration, followed by another full acceleration and final deceleration. This is due to the limitations in the steering rate: the maximum steering rate is not sufficient to allow the vehicle to smoothly turn and visit all three targets while also accelerating. Thus, in order to reduce time, the optimiser found a solution which is overcoming this limitation by reducing the velocity and thus reducing the turning radius. Low energy solutions instead have a smooth linear profile, corresponding to a progressive deceleration. Since the velocities are lower for these solutions, a smaller curvature radius is achievable, resulting in rectilinear trajectories for most of the interval connecting two targets.

Solution 10 can be compared with the solution found in the reference [16], which was solving the minimum time problem. The solution obtained by this algorithm is strictly better than the best solution reported in the reference. However it can be compared with one of the minimum time solutions belonging to the first layer of dominated solutions. Table 2 reports a comparison of the times when each solution visits each target and the total mission time. The difference in total mission between the reference and the "comparable" dominated solution is below 0.3% and can be attributed to the fact that the DFET transcription with Bernstein polynomials smooths the sharp discontinuities of bang-bang profiles, resulting in a slight penalisation of the objective function [9]. It is expected that with some iterations of mesh refinement, the bang-bang profiles of the solutions will be more sharp, thus decreasing the value of the mission time and evantually reaching the same value of the solution present in the literature. Solution 10 however is markedly better than the reference solution, and is also significantly different given that the trajectories computed by this run of the algorithm are all smooth, while solutions reported in the literature have a cusp around the target $P2$. The order in which the targets are visited is

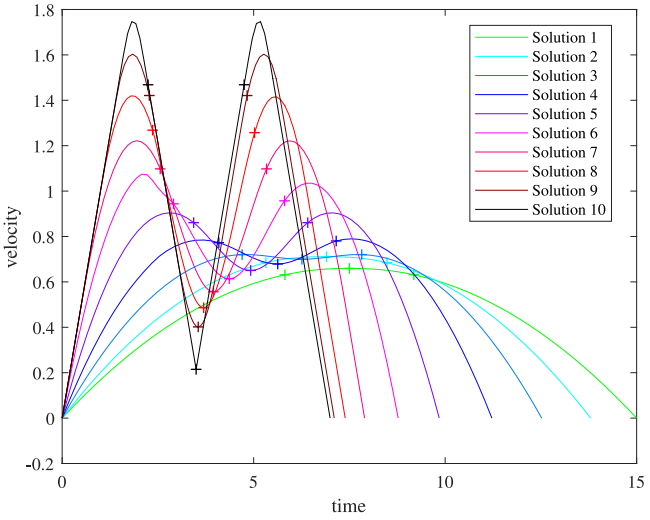

Figure 4: Time histories of velocity for the Travelling Salesman Problem. + indicate an encounter with a target

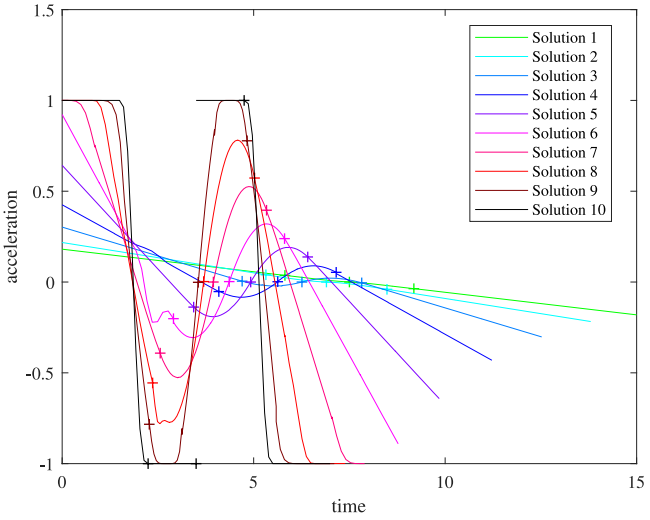

Figure 5: Time histories of acceleration for the Travelling Salesman Problem. + indicate an encounter with a target

also different: $P_1 \rightarrow P_2 \rightarrow P_3$ for the best solution found by the current algorithm vs $P_3 \rightarrow P_2 \rightarrow P_1$ for the best solution found by the algorithm in the literature, and for the "comparable" dominated solution.

An important fact is mentioned in reference [16]: multiple local solutions exist even for a given order in which the targets are visited. It was not specified how the different solutions were generated, but the authors mentioned that different initial guesses had to be provided to the NLP solver in order to obtain different solutions. Since the approach proposed in this work is global and treats simultaneously both the discrete and the continuous variables, this kind of exploration is performed automatically, and, as shown, is able

Table 2: Comparison of intermediate and final times between the reference solution, solution 10 on the Pareto front of the selected run, and a "comparable" solution in a dominated layer of another run

Solution	$T(P_1)$	$T(P_2)$	$T(P_3)$	T_f
Reference	5.3830 (s)	3.1390 (s)	2.286 (s)	7.6166 (s)
Comparable	5.3958 (s)	3.1475 (s)	2.296 (s)	7.639 (s)
Solution 10	2.2443 (s)	3.4980 (s)	4.7518 (s)	6.9960 (s)

to return even better solutions than those reported in the literature without requiring the user to generate different initial guesses for the NLP problem. Moreover, since the approach proposed in the reference was based on a deterministic branch and bound, it requires to potentially compute all the solutions at the leaf level of the tree associated to the branch and bound. If the number of targets increases, this number can become exceedingly large, resulting in intractable problems. With the approach proposed here, the maximum number of trials is specified a priori. Thus, even if larger problems become more and more difficult, the approach here proposed is still able to return a solution with a bounded number of function evaluations and computational time.

4.2 Multi target debris removal

As a final test case, the algorithm is applied to the design of a space mission to remove three debris. The debris are assumed to be 3 defunct satellites located on the Geostationary orbit. A spacecraft is sent to apply a de-orbiting kit on each of those satellites. The spacecraft is assumed to start from an equatorial orbit. Thus it is convenient to express its dynamics using the full nonlinear equations of relative motion in Hill's frame[11], restricted to a planar motion:

$$\begin{cases} \dot{x} = v_x \\ \dot{y} = v_y \\ \dot{v_x} = 2\dot{f}\left(v_y - y\frac{\dot{r_c}}{r_c}\right) + x\dot{f}^2 + \frac{\mu}{r_c^2} - \frac{\mu(r_c+x)}{r_d^3} + \frac{uT\cos(\alpha)}{m} \\ \dot{v_y} = -2\dot{f}\left(v_x - x\frac{\dot{r_c}}{r_c}\right) + y\dot{f}^2 - \frac{\mu y}{r_d^3} + \frac{uT\sin(\alpha)}{m} \\ \dot{m} = \frac{-uT}{g_0 I_{sp}} \end{cases} \tag{12}$$

where x and y are the positions of the spacecraft relative to a reference point on the Geostationary orbit, v_x and v_y are the relative velocities, $\dot{f}$ is the rate of change of true anomaly of the Geostationary orbit, r_c is the radius of the Geostationary orbit, r_d is the distance of the current position to the centre of the Earth and and m is the mass of the spacecraft, which is controlled by an engine with maximum thrust T and a specific impulse I_{sp}. The thrust vector is expressed in terms of throttling u and direction α. g_0 is the gravitational acceleration at sea level, while μ is the gravitational parameter of Earth.

The spacecraft has an initial mass of 1000 kg, a minimum mass of 100 kg and is equipped with 3 de-orbiting kits, each with a mass of 30 kg. The propulsion system is able to generate up to 10 N of thrust with a specific impulse of 300 s. The spacecraft starts from

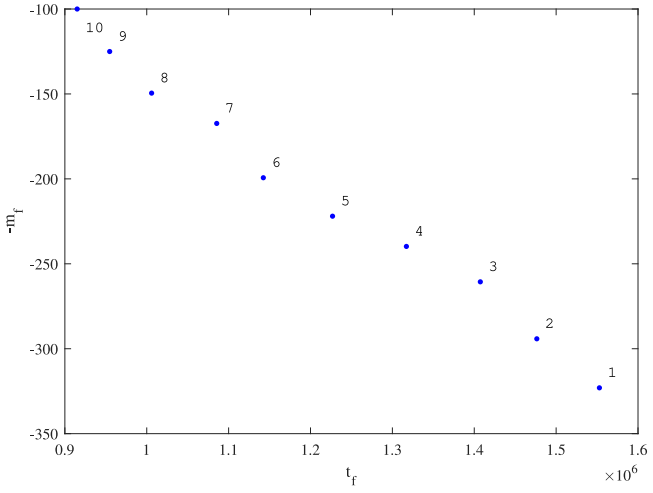

Figure 6: Pareto front for the multiple debris removal mission

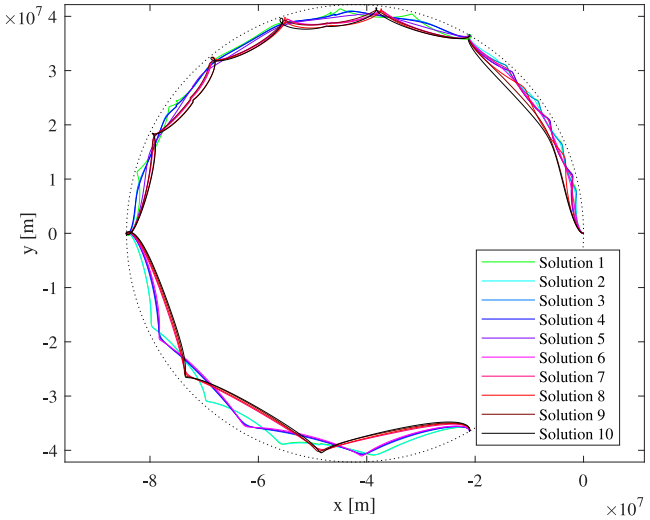

Figure 7: Trajectories for the multiple debris removal mission in the relative Hill's frame

an equatorial circular orbit with radius of 42 000 km, which is at a safe distance from the target orbit. This corresponds in the selected Hill's frame to the following initial conditions:

$$\begin{cases} x(0) = -164\,\text{km} \\ y(0) = 0\,\text{km} \\ v_x(0) = 0\,\text{m s}^{-1} \\ v_y(0) = 5.9971\,\text{m s}^{-1} \end{cases} \tag{13}$$

Since the reference frame is collocated on a point on the Geostationary orbit, the targets have a fixed position in time. This significantly simplifies the treatment of the boundary conditions, which no longer have a periodic time dependency over time. In this reference frame, they are assumed to be uniformly spread, forming an angle with the reference point on the Geostationary orbit of 30 deg, 180 deg and 300 deg. Since in this reference frame their positions are constant, their velocities are $0\,\text{m s}^{-1}$.

The spacecraft has to rendez-vous and dock with all targets in order to apply the de-orbit kit. The application of the de-orbiting kit is assumed to take 5 h. The order in which targets are to be visited is not specified a priori. The objectives are the minimisation of the total mission time, and the maximisation of the final mass, corresponding to the minimisation of propellent consumption:

$$\min_{t_f,\,\mathbf{u},\,\boldsymbol{\mu}} (J_1, J_2)^T = \left(t_f, -m_f\right) \tag{14}$$

The problem was formulated as a 3 phase problem, each phase was discretised with 3 DFET elements of order 7, resulting in a problem with 551 variables. The upper and lower bounds for state and control variables are: $-126\,492\,\text{km} \le x \le 42\,164\,\text{km}$, $-84\,328\,\text{km} \le y \le 84\,328\,\text{km}$, $-20\,\text{m s}^{-1} \le v_x \le 20\,\text{m s}^{-1}$, $-20\,\text{m s}^{-1} \le v_y \le 20\,\text{m s}^{-1}$, $100\,\text{kg} \le m \le 1000\,\text{kg}$, $0 \le u \le 1$, $-\pi\,\text{rad} \le \alpha \le \pi\,\text{rad}$, $0\,\text{d} \le t_f \le 40\,\text{d}$. The algorithm was run for 100000 function evaluations with standard settings, 10 agents storing 10 points on the Pareto front.

Figure 6 and 7 show the Pareto front and the trajectories of the spacecraft in the relative frame. A tradeoff between mission time and final mass is present, as expected. Faster trajectories tend to be closer to the centre of the circle, meaning that in order to reduce mission time the optimiser exploited the relative rotation rate between the two orbits. All targets are visited in the same order. The epicycloidal shape of the trajectories means that the trajectories are more eccentric, while more circular shaped arcs mean that the corresponding trajectory is closer to circular.

5 CONCLUSIONS

This paper presented a new method for solving dynamic travelling salesman problems in the form of Multi-Objctive Hybrid optimal control problems. It is based on the coupling of the DFET transcription with the MACS memetic multi-objective optimisation algorithm. The resulting discretised problem is solved with a combination of two approaches, a bi-level approach, which allows for global exploration and a generation of a well spread set of solutions, and a single-level approach, which guarantees local convergence. A staged relaxation approach has been employed when the NLP solver is invoked. The relaxed variables are then forced to assume integer values with a simple set of smooth constraints. A special set of constraints was proposed to deal with the choice of the targets in the Travelling Salesman problem. The approach was tested against a problem found in the literature, and against a multiple debris removal mission. For the first problem, the approach was able to generate better solutions than those reported in the literature without any user intervention. Future work will target more challenging space mission design scenarios, like multi gravity assisted interplanetary trajectories.

REFERENCES

[1] Michele Aicardi, Davide Giglio, and Riccardo Minciardi. 2008. Determination of optimal control strategies for TSP by dynamic programming. In *2008 47th IEEE Conference on Decision and Control*. IEEE, 2160–2167.

[2] Stephen M Akandwanaho, Aderemi O Adewumi, and Ayodele A Adebiyi. 2014. Solving dynamic traveling salesman problem using dynamic Gaussian process regression. *Journal of Applied Mathematics* 2014 (2014).

[3] Vassilis M Charitopoulos, Vivek Dua, and Lazaros G Papageorgiou. 2017. Traveling salesman problem-based integration of planning, scheduling, and optimal control for continuous processes. *Industrial & Engineering Chemistry Research* 56, 39 (2017), 11186–11205.

[4] Christian M Chilan and Bruce A Conway. 2013. Automated design of multiphase space missions using hybrid optimal control. *Journal of Guidance, Control, and Dynamics* 36, 5 (2013), 1410–1424.

[5] Markus Glocker and Oskar von Stryk. 2002. Hybrid optimal control of motorized traveling salesmen and beyond. In *Proc. 15th IFAC World Congress on Automatic Control*. 21–26.

[6] Ali Haghani and Soojung Jung. 2005. A dynamic vehicle routing problem with time-dependent travel times. *Computers & operations research* 32, 11 (2005), 2959–2986.

[7] Dario Izzo, Ingmar Getzner, Daniel Hennes, and Luís Felismino Simões. 2015. Evolving solutions to TSP variants for active space debris removal. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*. ACM, 1207–1214.

[8] Lorenzo A. Ricciardi, Christie Alisa Maddock, and Massimiliano Vasile. 2019. Direct Solution of Multi-Objective Optimal Control Problems Applied to Spaceplane Mission Design. *Journal of Guidance, Control, and Dynamics* 42, 1 (jan 2019), 30–46.

[9] Lorenzo A. Ricciardi and Massimiliano Vasile. 2018. Direct Transcription of Optimal Control Problems with Finite Elements on Bernstein Basis. *Journal of Guidance, Control, and Dynamics* (oct 2018), 1–15.

[10] Lorenzo A. Ricciardi and Massimiliano Vasile. 2018. Improved Archiving and Search Strategies for Multi Agent Collaborative Search. In *Computational Methods in Applied Sciences*. Springer International Publishing, 435–455.

[11] Hanspeter Schaub and John L. Junkins. 2018. *Analytical Mechanics of Space Systems, Fourth Edition*. American Institute of Aeronautics and Astronautics, Inc.

[12] Yong Song, Yongyuan Qin, Xianfu Chen, and Jinchuan You. 2009. Dynamic TSP optimization base on elastic adjustment. In *2009 Fifth International Conference on Natural Computation*, Vol. 4. IEEE, 205–210.

[13] Alejandro Toriello, William B Haskell, and Michael Poremba. 2014. A dynamic traveling salesman problem with stochastic arc costs. *Operations Research* 62, 5 (2014), 1107–1125.

[14] Massimiliano Vasile. 2010. Finite Elements in Time: A Direct Transcription Method for Optimal Control Problems. In *AIAA/AAS Astrodynamics Specialist Conference*.

[15] M. Vasile and F. Zuiani. 2011. Multi-Agent Collaborative Search: an Agent-based Memetic Multi-objective Optimization Algorithm Applied to Space Trajectory Design. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 225, 11 (2011), 1211–1227.

[16] O Von Stryk and M Glocker. 2001. Numerical mixed-integer optimal control and motorized traveling salesmen problems. Commande numerique optimale a entiers mixtes et problemes de voyageur de commerce motorise. *Journal Européen des Systèmes Automatisés* 35, 4 (2001), 519–534.

[17] Bradley J Wall and Bruce A Conway. 2009. Genetic algorithms applied to the solution of hybrid optimal control problems in astrodynamics. *Journal of Global Optimization* 44, 4 (2009), 493.

[18] F. Zuiani and M. Vasile. 2013. Multi-Agent Collaborative Search Based on Tchebycheff Decomposition. *Computational Optimization and Applications* (Mar. 2013). https://doi.org/10.1007/s10589-013-9552-9